ALGORITHMES SANS BIAIS
DE
LIGNE DE PARTAGE DES EAUX

Serge BEUCHER
Centre de Morphologie Mathématique
Ecole des Mines de Paris

28 Février 2002
Révisé le 30 Avril 2004

TABLE DES MATIERES

ALGORITHMES SANS BIAIS DE LIGNE DE PARTAGE DES EAUX

Serge BEUCHER
Centre de Morphologie Mathématique
Ecole des Mines de Paris

28 Février 2002
Révisé le 30 Avril 2004

Avant-Propos

Cette version révisée de cet article permet de préciser certains points et de corriger quelques erreurs et imprécisions apparaissant dans l'algorithme sans biais de ligne de partage des eaux par files d'attente hiérarchiques décrit dans la première version. Ces erreurs étaient dues à une gestion imparfaite des parités lors de la propagation sur les plateaux.

Les modifications apportées à l'algorithme initial de LPE par FAH sont similaires à celles qui avaient été proposées intialement pour paralléliser l'algorithme. L'algorithme parallèle que je décris dans la troisième partie du document n'est pas encore définitif. Il mériterait d'être simulé et validé. Cependant, il constitue une base de travail solide pour la définition d'une architecture appropriée. Le besoin d'une telle architecture se fait de plus en plus sentir à un moment où, d'une part de nouveaux outils de segementation associés à la LPE apparaissent et où, d'autre part, de nouveaux domaines d'applications voient le jour (vidéo-surveillance, caméras intelligentes).

Introduction

L'utilisation de files d'attente hiérarchiques (FAH) apparaît actuellement comme la meilleure solution pour construire rapidement la ligne de partage des eaux d'une image. En effet cet algorithme, basé sur un processus d'inondation similaire à celui utilisé dans la définition opératoire de la LPE, est cependant beaucoup plus rapide que l'algorithme classique utilisant des opérateurs de morphologie (SKIZ géodésique et reconstruction) niveau par niveau. De plus, cet algorithme gère très bien la LPE contrôlée par marqueurs.

Néanmoins, cette FAH est entachée du même défaut que l'algorithme classique: la LPE produite est biaisée, ce qui entraîne un certain nombre de difficultés lors de l'utilisation de cette transformation.

Cette note est divisée en trois parties. La première (et la plus importante) sera consacrée à la description d'un algorithme de LPE par FAH sans biais. Je rappellerai d'abord le fonctionnement de la FAH et j'expliciterai la nature des biais et leur ampleur. Puis je décrirai un algorithme qui, sans aucunement modifier la structure de la FAH, n'introduit aucun biais. Dans la deuxième partie, un algorithme sans biais basé sur les mêmes principes mais utilisant l'approche classique sera décrit. On verra que cet algorithme produit une ligne de partage des eaux absolument identique à celle obtenue par l'algorithme de FAH sans biais. L'avantage de cet algorithme est uniquement de fournir une solution au problème de la

réalisation d'une ligne de partage des eaux sans biais lorsqu'on ne dispose que d'outils morphologiques standards. Dans la troisième partie enfin, je discuterai des perspectives offertes par ces algorithmes sans biais en matière de parallèlisation.

1ère partie: une ligne de partage des eaux exacte par FAH

1.1) Rappel: l'algorithme de LPE par files d'attente hiérarchiques

Commençons par rappeler le fonctionnement de l'algorithme par files d'attente hiérarchiques. Il en existe plusieurs variantes. Seul l'algorithme générant les bassins versants sans frontières sera pris en compte ici. Le lecteur pourra se reporter à [7] pour plus de détails concernant ces diverses variantes.

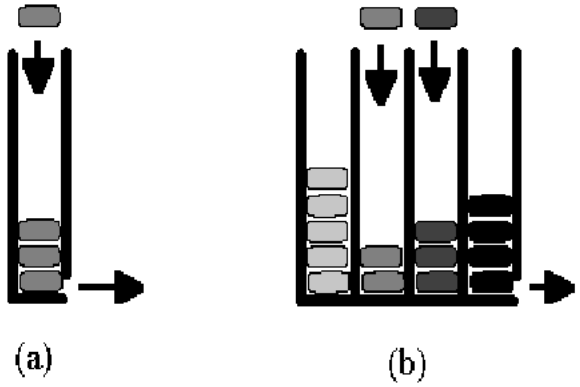


Figure 1: File d'attente simple (a) et file d'attente hiérarchique (b)

Une file d'attente hiérarchique est l'assemblage de N files d'attentes simples (figure 1a). Une file d'attente est encore appelée registre FIFO ("First In First Out"). Les jetons en sont extraits dans l'ordre chronologique de leur introduction (figure 1b). Chaque file d'attente simple a un niveau de priorité. Ce niveau de priorité correspond au niveau de gris du jeton (pixel). Les pixels de plus faible niveau de gris ont la priorité la plus forte (les files sont numérotées selon le niveau de gris; la file 0 a donc la plus forte priorité). Le nombre N correspond donc au nombre de niveaux de gris de l'image. Toutes les files sont ouvertes à leur sommet: à tout moment un jeton peut être inséré dans la file de priorité correspondante. Cependant, seule la file de plus forte priorité peut être vidée: le jeton extrait est celui de plus forte priorité arrivé le premier dans la file (figure 1b). Enfin, dès que la file de plus forte priorité est vide, elle est supprimée et la file d'attente suivante peut alors commencer à se vider. Cette dernière règle de fonctionnement est importante car elle assure la progression séquentielle niveau par niveau de l'inondation. Ce mode de fonctionnement est fondamental dans le cas de la LPE contrôlée par marqueurs car il réalise en quelque sorte une modification d'homotopie "à la volée". Mais il est tout aussi critique pour gérer correctement l'inondation de certaines structures dans la LPE simple (en particulier les boutonnières, cf. [1] et [5]).

Deux images sont associées à la FAH. La première est l'image f elle-même. Cette image définit le nombre de files d'attente et les priorités (niveaux de gris) des jetons. La deuxième image, appelée image-label et notée g contient les étiquettes des pixels traités.

Chaque bassin versant est étiqueté avec un label strictement positif. L'image-label indique donc au cours du processus à quel bassin versant de la LPE appartient chaque pixel. Un pixel de label 0 est par définition un pixel pas encore affecté à un bassin versant.

L'image-label g est initialisée avec des valeurs correspondant aux labels des différentes sources d'inondation (minima ou marqueurs). La FAH est initialisée en stockant dans les files d'attente respectives les jetons correspondant aux pixels étiquetés dans l'image g. Ces jetons transportent une seule information: les coordonnées du pixel correspondant dans f ou g. L'ordre d'introduction des jetons dans une file d'attente est quelconque. Il correspond généralement à l'ordre de balayage de l'image. On peut aussi, pour gagner du temps, se contenter d'empiler les jetons correspondant à la frontière des marqueurs.

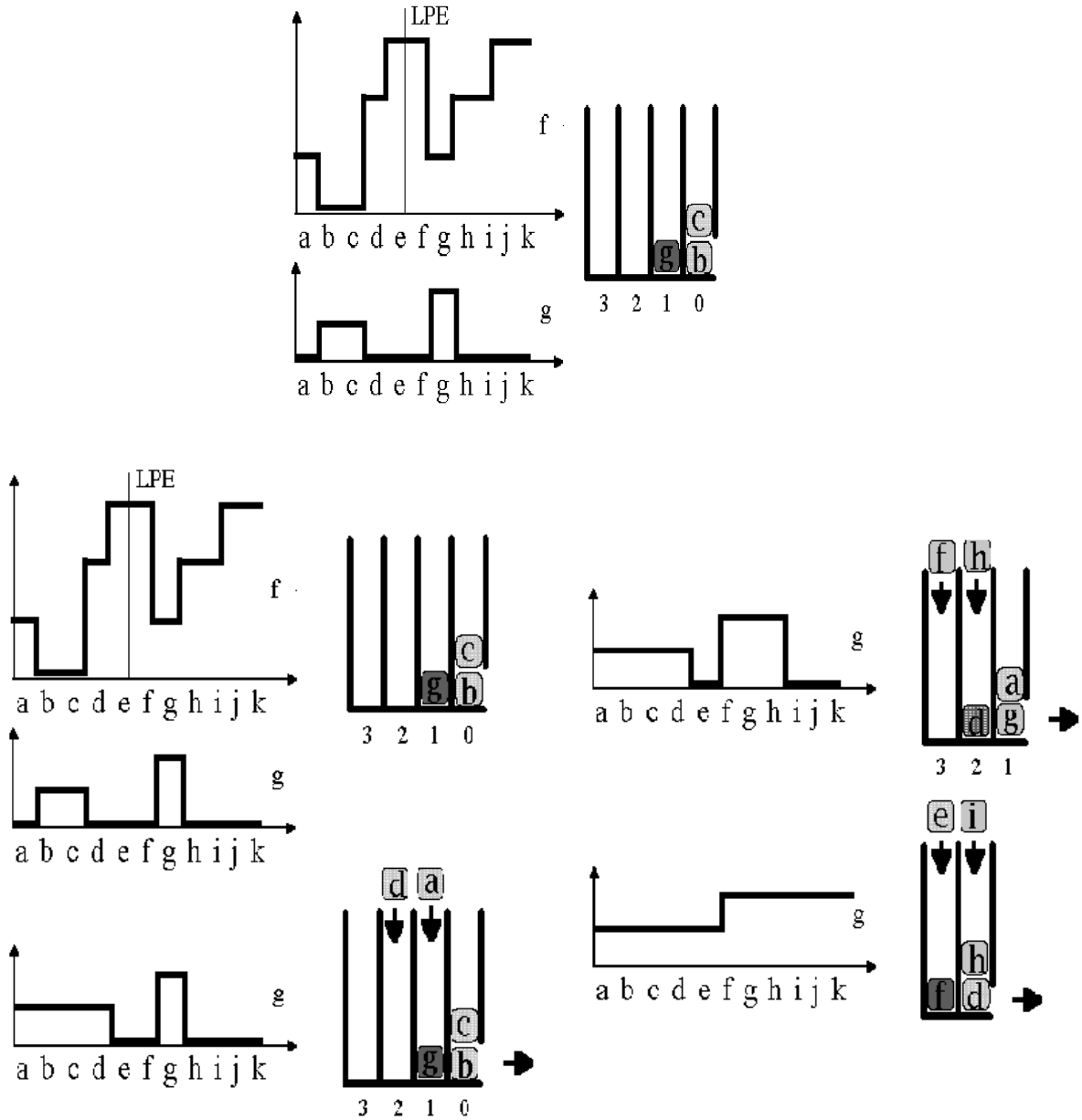


Figure 2: les trois éléments de la structure: image, image-label et FAH (en haut)
Les étapes successives de l'algorithme (en bas)

Le fonctionnement de la FAH est alors contrôlée par l'algorithme suivant appliqué à chaque jeton:

Tant que la FAH n'est pas vide, faire:
{- extraire un jeton x de la FAH
- déterminer les pixels voisins de x d'étiquette nulle dans g
- pour chaque voisin y d'étiquette nulle, faire:
{ - assigner à y dans l'image g la même étiquette que x.
- insérer le jeton y dans la file d'attente de la FAH de priorité correspondant au niveau de gris de y dans f (si elle existe) ou à la file d'attente de plus forte priorité existant encore. }
}

La figure 2 ci-dessus illustre ce fonctionnement dans un cas simple (monodimensionnel).

Rappelons l'importance de la règle de gestion des files disparues. Il peut arriver, en effet, que des jetons de priorité inférieure apparaissent alors que la pile correspondante n'existe plus. Dans ce cas, le jeton est inséré dans la pile courante (figure 3). Cette situation se produit fréquemment dans la réalisation de la LPE contrôlée par marqueurs *mais pas seulement*. La figure 3 montre le fonctionnement de la FAH dans le cas où l'initialisation se fait avec des marqueurs quelconques et non avec les minima de f. On voit effectivement apparaître, dans ce cas, des jetons de priorité plus élevée que celle de la file courante. Tout se passe alors comme si la valeur de gris du pixel correspondant était rehaussé à la valeur correspondant à la hauteur courante de l'inondation, ce qui constitue, en quelque sorte, une modification d'homotopie "à la volée". Cependant, je le répète, cette règle est aussi indispensable dans le cas de la construction de la LPE simple où il peut arriver que des points de hauteur inférieure à celle de l'inondation courante n'aient pas encore été pris en compte (cas des boutonnières notamment, structures fréquemment rencontrées dans les images réelles).

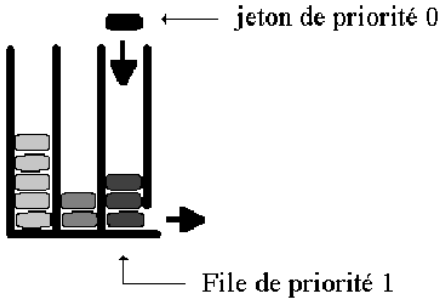


Figure 3: règle de gestion des jetons de priorité inférieure

1.2) Les biais de l'algorithme

L'algorithme de LPE par FAH tel qu'il a été décrit précédemment est malheureusement biaisé. En effet, il ne reproduit pas exactement le processus d'inondation défini dans la LPE. Si ce processus se déroule effectivement niveau par niveau, ce que le traitement séquentiel des diverses files d'attente simule très bien, en revanche dans la réalité, à chaque niveau, tous les pixels adjacents aux zones inondées sont traités *en même temps*, en parallèle, ce qui n'est pas le cas dans l'algorithme de LPE par FAH. En effet, chaque jeton est

traité dans un ordre arbitraire, celui qui a été choisi pour les empiler. Il en résulte alors une mauvaise affectation de certains pixels de l'image à un bassin versant. Ce biais peut se remarquer sur l'exemple suivant (figure 4): le pixel (g) appartenant à la ligne de partage des eaux est en fait affecté au bassin versant n° 2 simplement parce (g) a été empilé et affecté à ce bassin versant lors du traitement de (h). Or (g) est également inondé par (f). Seulement (f) étant à une altitude plus élevé que (h), il est traité après. Quand son tour arrive, son voisin (g) a déjà été étiqueté. Le même phénomène se produit avec le pixel (j).

De façon plus précise, le biais provient donc du fait que les voisins d'un jeton sortant de la FAH sont également les voisins d'autres jetons pas encore traités. Mais à ce biais s'ajoute un autre biais provenant de l'ordre de traitement des voisins d'un jeton. En effet, cet ordre est arbitraire et engendre donc des biais similaires à ceux rencontrés lorsqu'on construit les squelettes à partir d'amincissements ou d'épaississements en rotation [8]. C'est d'ailleurs pour cette raison que l'algorithme classique de LPE basé sur le SKIZ géodésique est entaché des mêmes biais. Le SKIZ, en effet, est réalisé avec des épaississements en rotation qui traitent les pixels voisins selon un ordre arbitraire. Ce deuxième biais apparaît lors de l'inondation des zones plates. En effet, ce n'est que dans ce cas que le choix des pixels voisins du pixel courant est arbitraire car ces pixels voisins ont la même hauteur que le pixel courant.

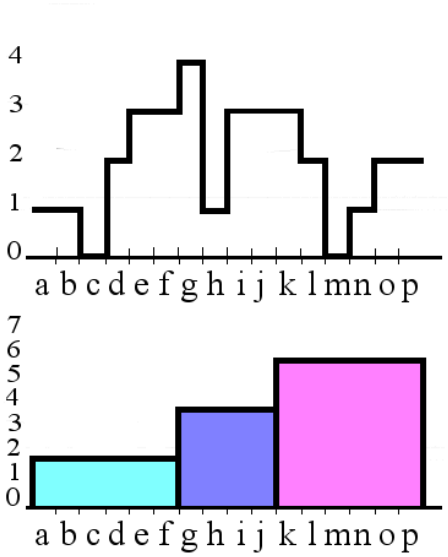


Figure 4: biais de la LPE

L'exemple précédent pourrait laisser croire que ce biais n'est pas très important, qu'il engendre seulement un décalage d'un ou deux pixels par rapport à la position réelle de la LPE. Ce serait là une profonde erreur. Ce biais peut en fait être considérable car les erreurs se propagent. La figure 5 montre les écarts importants pouvant exister entre une LPE vraie (à gauche) et la même transformation réalisée avec un algorithme de FAH (dans les deux cas la ligne de partage des eaux est matérialisée). L'ampleur du biais est encore plus marquée dans la LPE tridimensionnelle. Ce biais a d'ailleurs été souvent mis en évidence par les utilisateurs de la ligne de partage des eaux. Ils parlent souvent de "fuite", attribuant à tort ce phénomène à la LPE elle-même alors qu'il est dû en fait à l'*algorithme* utilisé pour la générer.

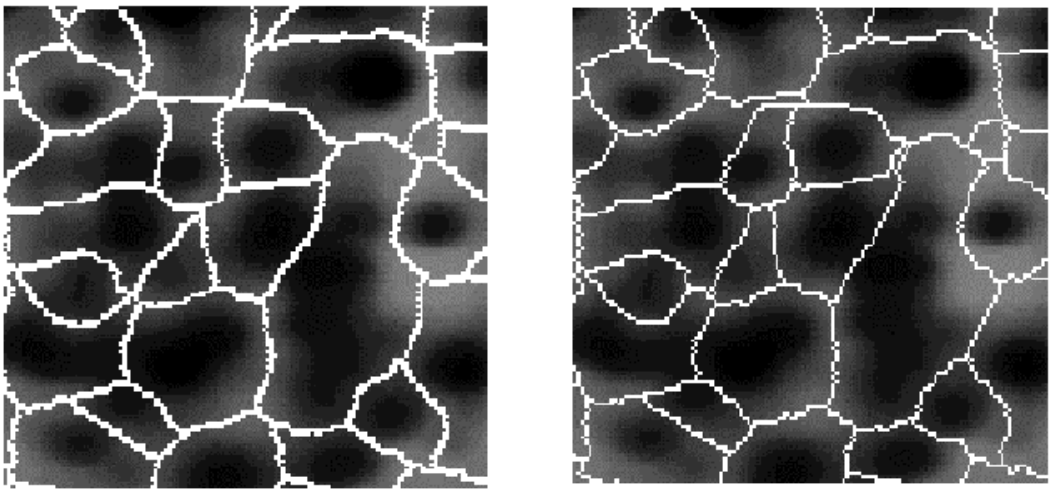


Figure 5: LPE vraie (à gauche) et LPE biaisée (à droite)
Les "fuites" sont particulièrement visibles

Pourquoi ce biais est-il gênant? Les raisons sont diverses. La première provient du fait que la LPE n'est pas unique. Autant d'implémentations, autant de résultats différents. Cela conduit à considérer la LPE comme une transformation clôturant un processus plutôt que comme une étape intermédiaire dans une chaîne d'algorithmes. Imaginons en effet que les transformations morphologiques de base comme l'érosion ou la dilatation présentent le même type de biais. Il serait alors difficile de les combiner pour définir des opérateurs fiables, car reproductibles. Une transformation complexe comme l'érodé ultime, un filtre morphologique, etc., produirait, sur la même image, des résultats très différents si elle était réalisée sur des systèmes différents. Notons que c'est précisément ce qui se produit actuellement avec des transformations comme le squelette lorsqu'elles sont réalisées à l'aide d'amincissements en rotation. Et on sait le faible intérêt que suscite ces squelettes comme descripteurs ou comme comparateurs de forme...

La même désaffection risque de se produire avec la LPE. Or, cette transformation est de plus en plus utilisée comme opérateur de bas niveau dans des algorithmes de segmentation complexes: hiérarchies et/ou segmentation multi-critères [2]. Ces approches nécessitant la comparaison de différentes lignes de partage des eaux ou étant basées sur les relations de voisinage entre bassins versants, on imagine aisément les problèmes soulevés par l'utilisation d'algorithmes biaisés et la qualité finale des résultats obtenus.

1.3) Un algorithme de LPE par FAH sans biais

L'algorithme suivant est un algorithme sans biais de LPE par FAH. Il conserve intégralement la structure de la FAH ainsi que la mémoire-label associée. Seules trois modifications sont effectuées:

- les règles d'étiquetage des jetons dans la mémoire-label sont modifiées. De plus la valeur des pixels correspondants dans la mémoire-image peut éventuellement être modifiée provisoirement (on précisera ce point).
- une file d'attente intermédiaire est ajoutée. Cette file d'attente permet de gérer de façon correcte la propagation de l'inondation sur les zones plates.

- trois types de labels différents sont introduits: des labels définitifs, des labels provisoires et un label indiquant une “collision” d’inondations provenant de bassins versants différents.

On a vu que les biais proviennent essentiellement du fait que les voisins pas encore étiquetés du jeton sortant prennent le label de ce dernier sans avoir vérifié au préalable que ledit voisin ne soit pas susceptible d’être également inondé à partir d’un autre bassin versant. Cette situation appelée collision est illustrée à la figure 6.

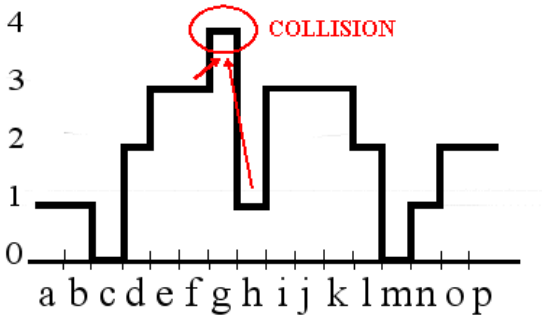


Figure 6: situation de collision (cas monodimensionnel)

Pour gérer cette situation, une solution consiste à affecter à chaque pixel voisin du jeton en cours de traitement et qui n’est pas encore étiqueté, un label provisoire. Ce label est relié au label du jeton sortant de façon biunivoque. Ainsi, dans l’algorithme présenté ici, les labels définitifs sont des labels pairs. Les labels provisoires sont les labels impairs immédiatement inférieurs. De cette façon, un jeton de label pair p transmettra le label p-1 à un voisin non étiqueté.

Si, par contre, le pixel voisin est déjà étiqueté (avec un label provisoire), soit ce label est égal à p-1 et dans ce cas rien ne change (il est toujours inondé par le même bassin versant), soit il est différent. Nous sommes alors dans ce dernier cas dans une situation de collision (le voisin reçoit les eaux en provenance d’au moins deux bassins versants différents). Le label du voisin est alors remplacé par un label spécial indiquant cette situation.

Cependant, la gestion des collisions doit être effectuée avec prudence sur les zones plates. Il se peut en effet que des collisions soient engendrées sur ces zones plates alors qu’elles n’ont pas lieu d’être car elles se produisent entre des pixels appartenant à la même génération d’inondation. Ces collisions sont la conséquence de phénomènes de parité. Illustrons cela à l’aide d’un exemple.

La figure 7 montre la propagation sur deux plateaux (ici encore, l’exemple est mono-dimensionnel, mais cela n’enlève rien à son caractère illustratif). Deux cas de figure sont illustrés selon que le plateau présente un nombre pair ou impair de points. On constate que les jetons (b) et (d) marquent le jeton (c) en collision. Les jetons (f) et (i) stockent respectivement les jetons (g) et (h) sur la file d’attente courante en affectant aux pixels correspondants les labels provisoires correspondants. Il n’y a, à ce stade de la propagation, aucune erreur. Cependant, lorsque le jeton (g) sort de la pile, il va marquer le pixel (h) comme un pixel-collision. En effet, ce pixel voisin de (g) est affecté d’un label provisoire différent du label provisoire correspondant à (g). Ce marquage est erroné puisqu’on a vu que

le pixel (h) appartenait sans ambiguité aucune au bassin versant correspondant au minimum (j).

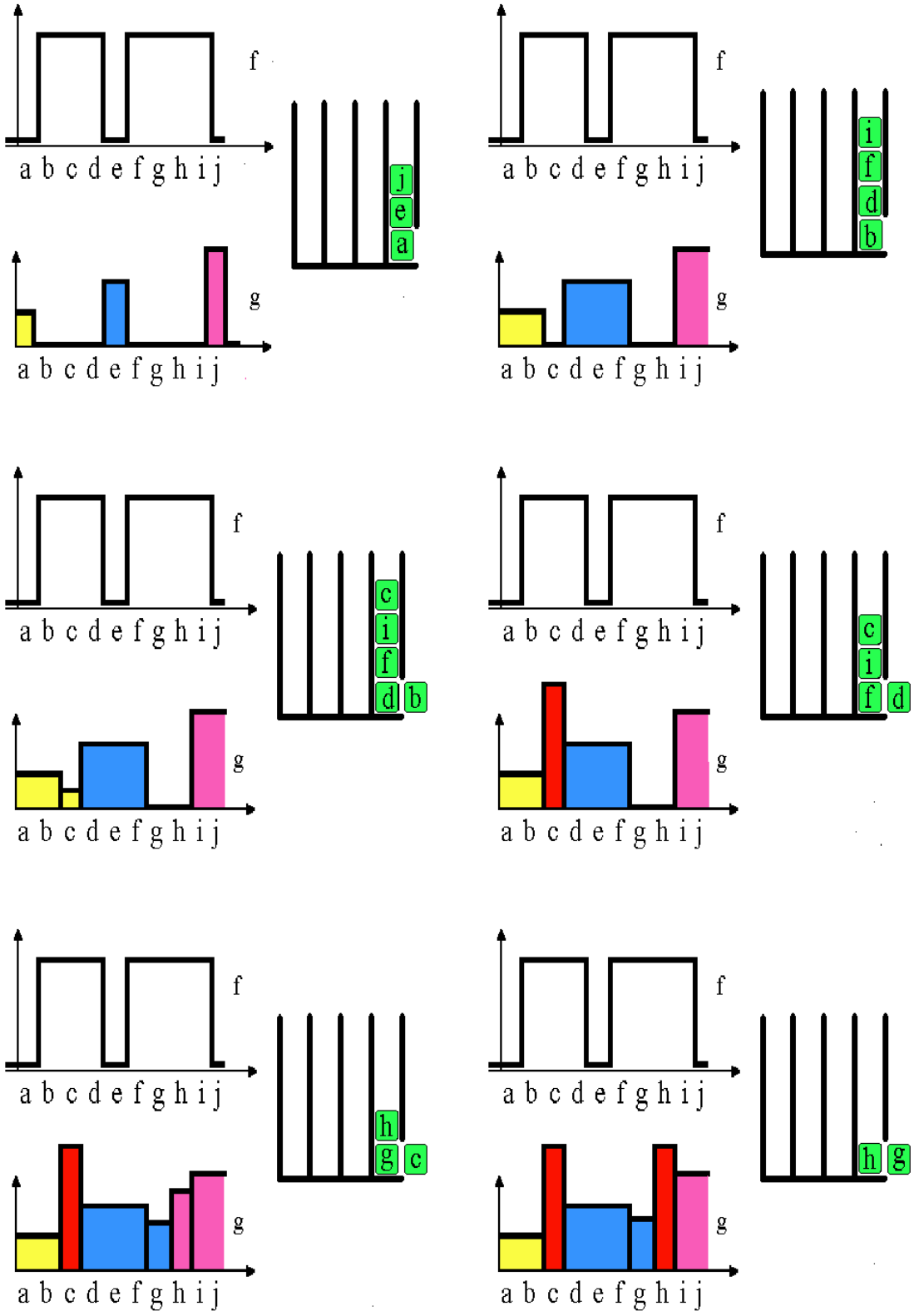


Figure 7: Gestion erronée des collisions sur les plateaux

Essayons de mettre en évidence ce qui distingue ces deux cas. Pour cela, rappelons la façon dont la propagation de l'inondation se fait sur les zones plates. Celle-ci s'effectue à vitesse constante en partant des bords descendants des plateaux. Il y a plusieurs générations (fronts) de propagation. La collision du point (c) est légitime car elle est engendrée par les points (b) et (d) qui appartiennent à la même génération de propagation alors que le point (c) appartient à la génération suivante. Inversement, la collision générée au point (h) est erronée car elle affecte un point qui fait partie de la génération courante de propagation et elle est partiellement engendrée par le point (g) qui fait, lui aussi, partie du front courant de propagation.

Pour résoudre ce dilemme, il est indispensable de séparer les différentes générations de jetons apparaissant dans les zones plates. Certes, l'implantatation courante ne les mélange pas puisque les jetons de génération n+1 sont toujours au-dessus des jetons de génrération n dans la pile courante. Cependant, il est impossible de savoir quand on change de génération. Une solution peut consister à mettre une séparation (un taquet) entre les différentes générations de jetons (voir plus loin). On peut également (ce qui revient au même) insérer une pile intermédiaire entre la pile courante et la pile de priorité suivante. La priorité de cette pile est notée α . Mais cette modification n'est pas suffisante. Il faut également indiquer que les pixels correspondant aux jetons insérés dans la pile intermédiaire appartiennent à la génération suivante de propagation de l'inondation sur les zones plates. Cela peut se faire en modifiant la valeur de ces pixels dans la mémoire-image. On peut remplacer la valeur courante, égale à la priorité k de la pile courante par une valeur quelconque différente des valeurs prises par les pixels de l'image. Si l'image est définie dans l'intervalle [0,N-1], une solution consiste à prendre la valeur N+n, n représentant le numéro de la pile α courante, plusieurs piles α successives pouvant être engendrées lors de la propagation sur un plateau. La valeur n correspond d'ailleurs à la valeur de la distance géodésique des points des zones plates par rapport aux bords descendants. Cette façon de faire est en fait nécessaire car on a besoin, lorsqu'on dépile la pile courante, de savoir à quelle génération de propagation appartiennent les jetons extraits.

Le diagramme ci-après (Figure 8) décrit de façon exhaustive l'algorithme de gestion des jetons extraits de la file d'attente. Dans cette réalisation de l'algorithme, on supposera que l'image f prend ses valeurs dans l'intervalle [0,N] et que la mémoire-label g permet d'étiqueter M bassins versants. Le label C correspondant aux collisions sera alors égale à 2M+1.

On dénote par k la priorité de la pile courante (correspondant à la hauteur courante des pixels traités) et par n la génération de propagation (correspondant à la n-ième génération de pile α). Les jetons correspondants aux minima ou aux marqueurs sont stockés dans leurs piles respectives et prennent un label différent et provisoire. La gestion des jetons qui devaient appartenir à des piles disparues ne change pas: ils sont simplement stockés dans la pile α .

Lorsque la pile courante est vide, la pile intermédiaire devient la pile courante et une nouvelle pile intermédiaire est insérée entre cette nouvelle pile courante qui conserve la priorité courante et la pile de priorité supérieure (toutes ces modifications se résument en fait à des modifications de pointeurs, il n'y a pas vidage effectif des piles, ce qui prendrait inutilement trop de temps). On incrémente alors la valeur n. Enfin, lorsque la pile intermédiaire est vide, la pile de priorité supérieure est traitée à son tour, en veillant à ce

qu'une pile intermédaire soit insérée juste après cette dernière. On met à jour k avec la valeur de priorité de cette nouvelle pile et n est remis à zéro.

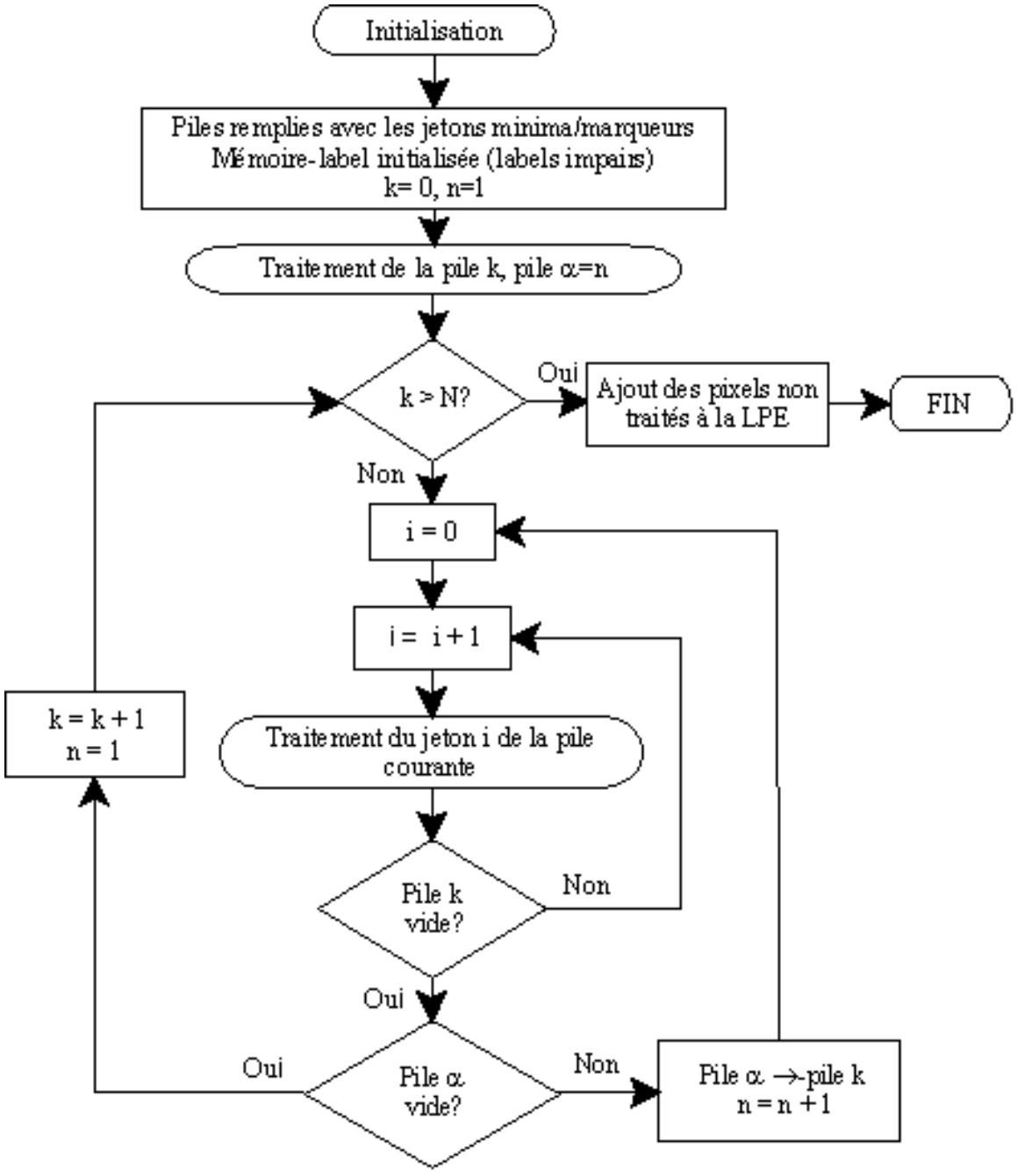


Figure 8: (a) Organigramme général de la LPE par FAH sans biais

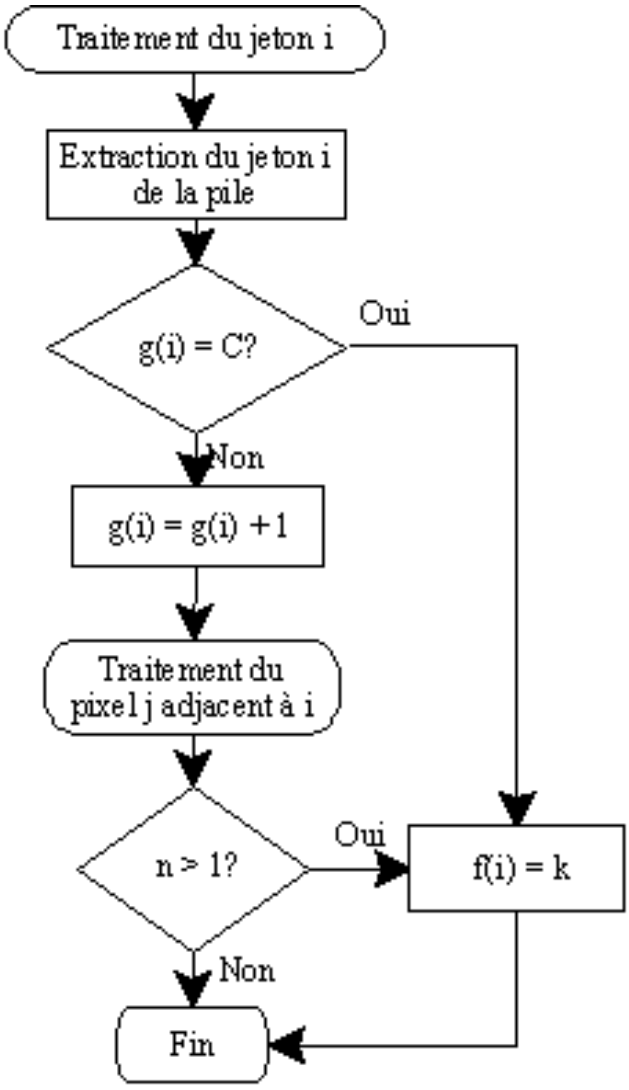


Figure 8 (suite): (b) Traitement du jeton i de la file d'attente

Deux points doivent être soulignés. Tout d'abord, chaque jeton ne prend son label définitif, et ce de façon systématique, qu'en sortie de file d'attente. Le changement de parité du label correspond au changement de statut du pixel qui passe de l'état de pixel inondé à celui de pixel inondant (dans un contexte de propagation, on dirait que, de fléché, le pixel devient flécheur). Ce traitement isotrope évite donc la génération des biais. Ensuite, un pixel en collision sort de la file sans propager d'inondation. Cette deuxième règle, tout aussi importante, évite la propagation des biais, à l'inverse de ce qui se produit dans l'algorithme de FAH classique.

Les figures 9a à 9f ci-après illustrent le fonctionnement de l'algorithme. La figure 10 est un exemple supplémentaire montrant le fonctionnement des piles α sur les plateaux.

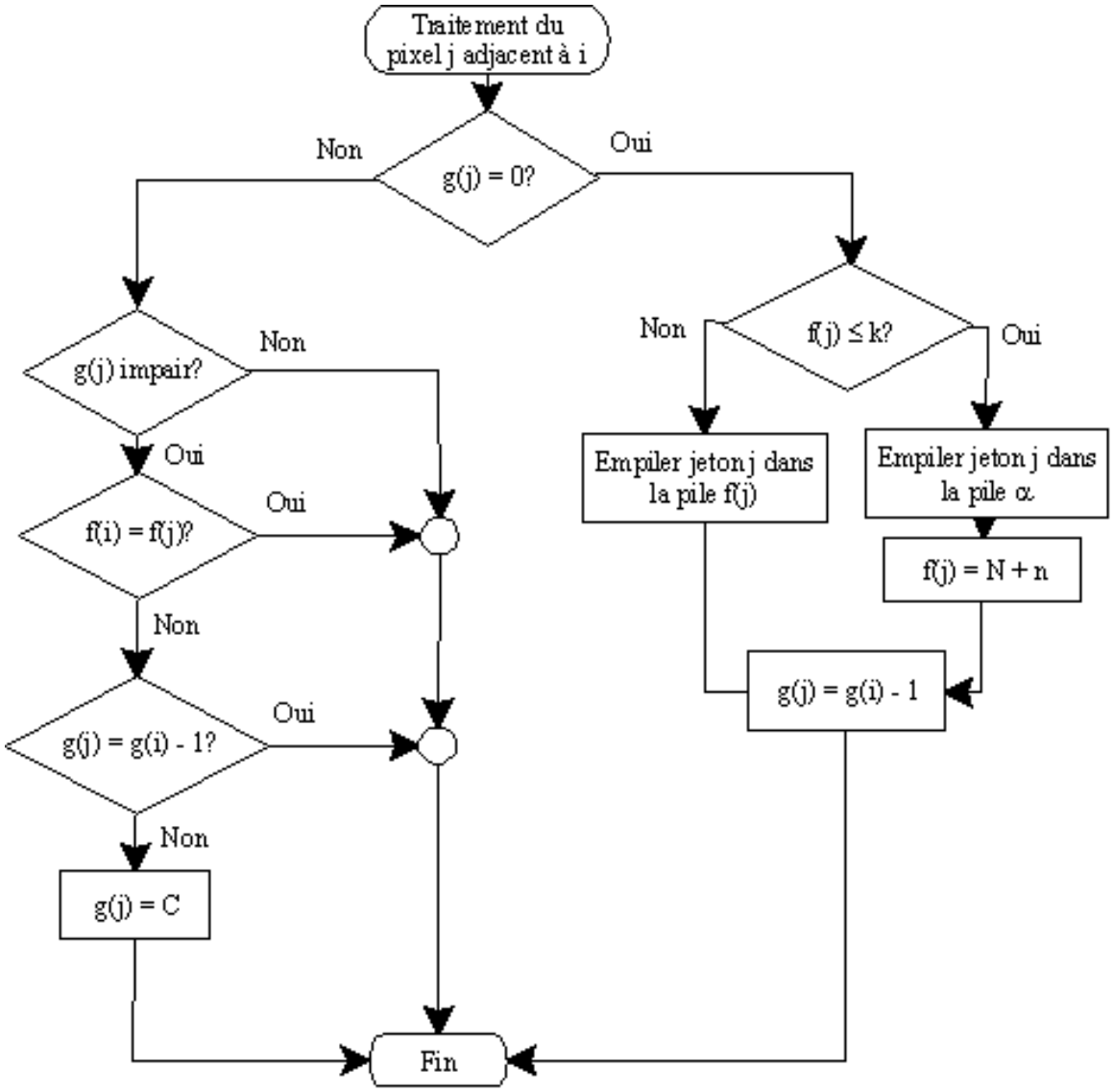
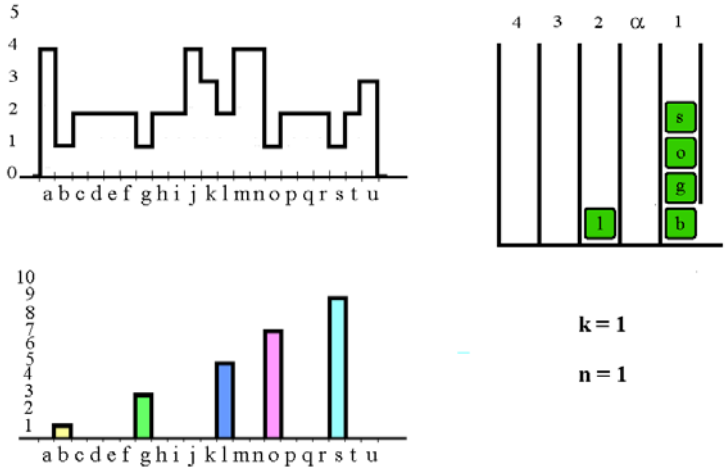
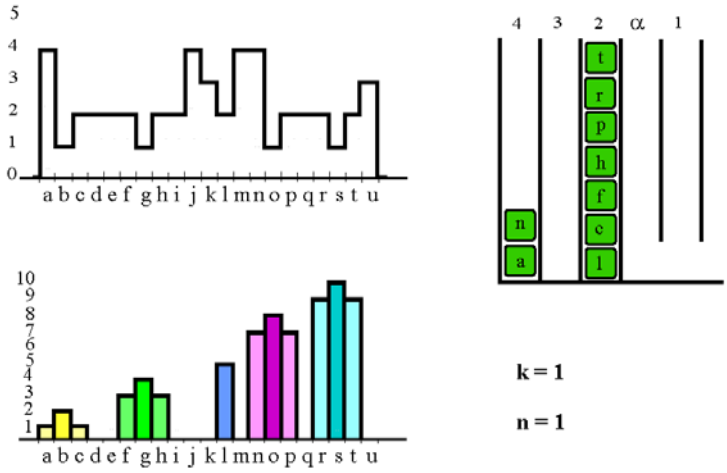


Figure 8: (c) Traitement des voisins du jeton courant

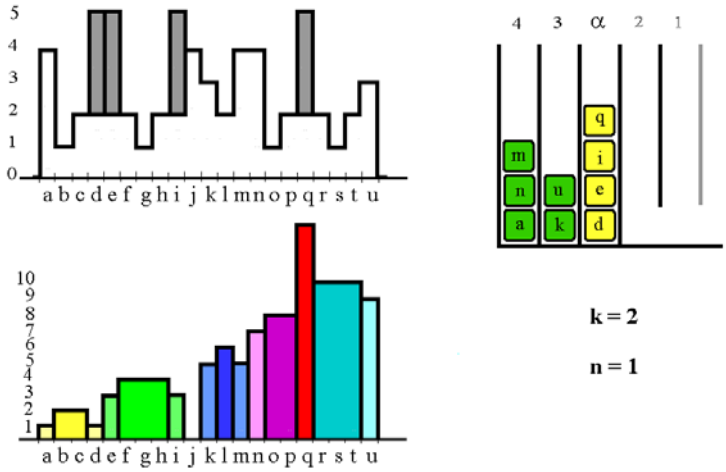
D'autres implantations de l'algorithme peuvent être envisagées. On pourrait remplacer la pile α par un taquet dans la pile courante, voire un jeton spécifique indiquant le changement de front de propagation. On pourrait également remplacer la modification de la mémoire-image par un ensemble de deux mémoires drapeaux marquant les pixels appartennat aux deux fronts de propagation successifs. En effet, il n'est pas nécessaire de distinguer tous les niveaux de propagation sur les plateaux. Seulle la distinction entre le niveau courant et le niveau suivant est indispensable. Cela pourrait alors se faire par le biais de deux drapeaux qu'on intervertit au fur et à mesure des propagations successives. On peut constater aussi que l'algorithme décrit ci-dessus, s'il gère parfaitement la LPE contrôlée par marqueurs, modifie en revanche la mémoire-image puisque, en sortie, les valeurs de certains pixels peuvent être remplacées par la hauteur courante de l'inondation. L'algorithme génère la modification d'homotopie correspondante en même temps qu'il construit la LPE.



(a)

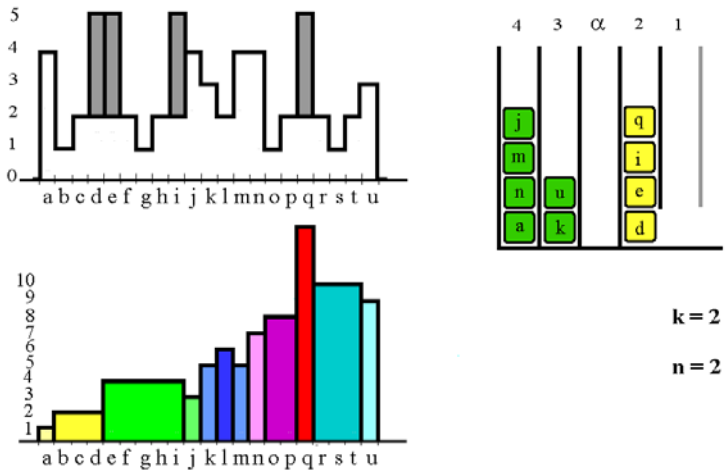


(b)

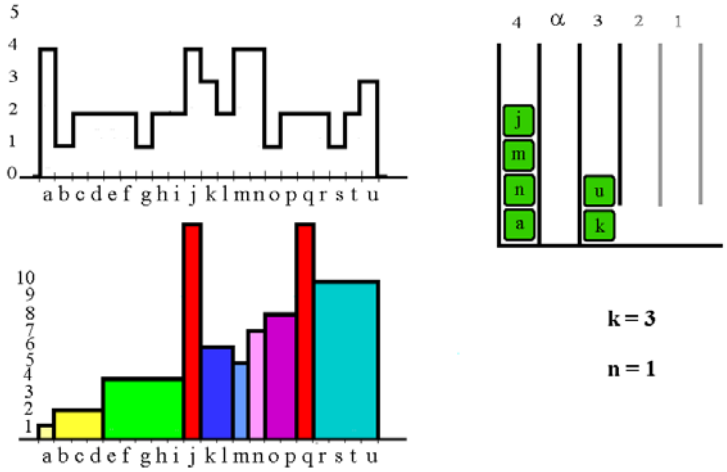


(c)

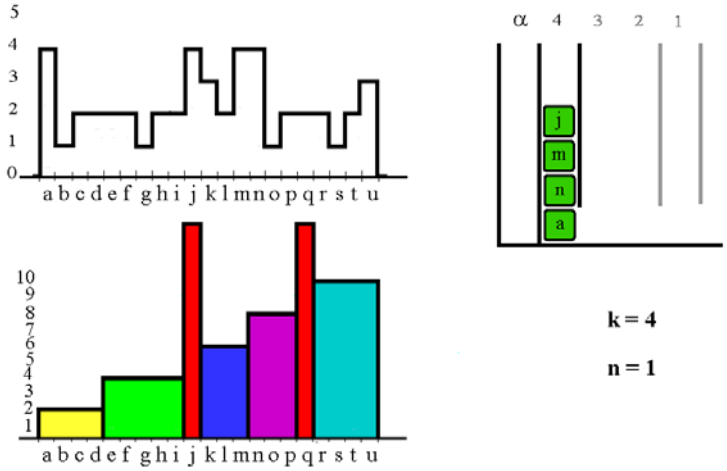
Figure 9: étapes successives de la FAH sans biais (début)
Les jetons empilés dans la pile α sont en jaune
Les labels définitifs sont pairs, les provisoires impairs



(d)



(e)



(f)

Figure 9: étapes successives de la FAH sans biais (fin)
Les jetons de la pile α sont transférés dans la pile courante

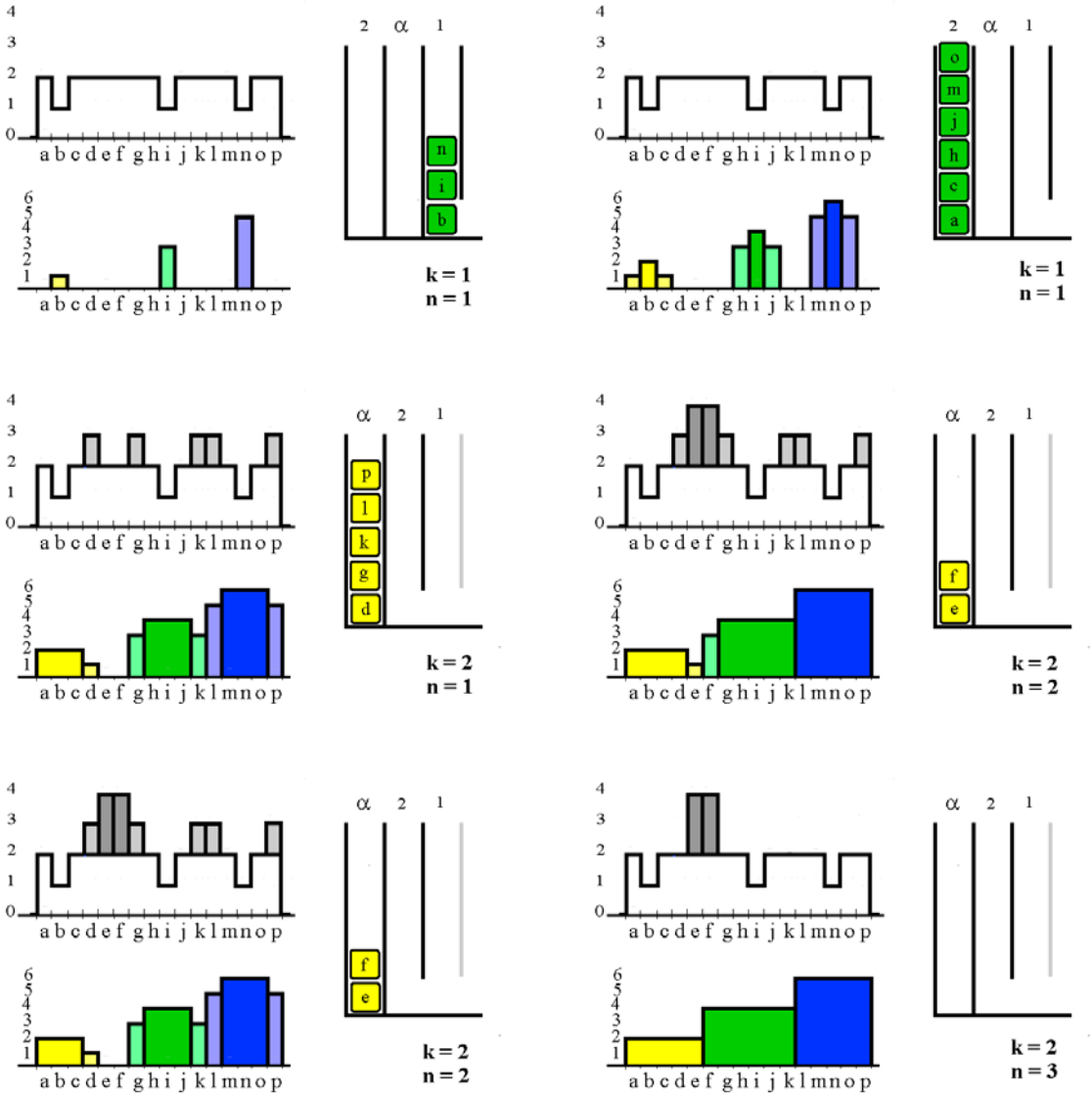
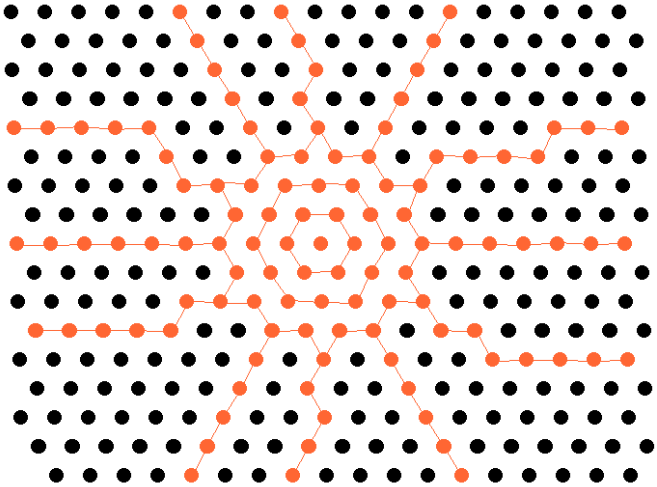
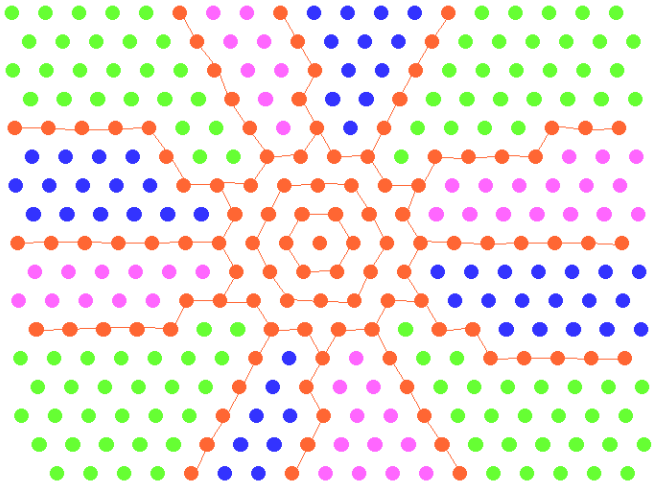


Figure 10: Fonctionnement des piles α sur les plateaux.
La fonction f utilise seulement deux valeurs différentes pour indiquer les générations de jetons à chaque étape.

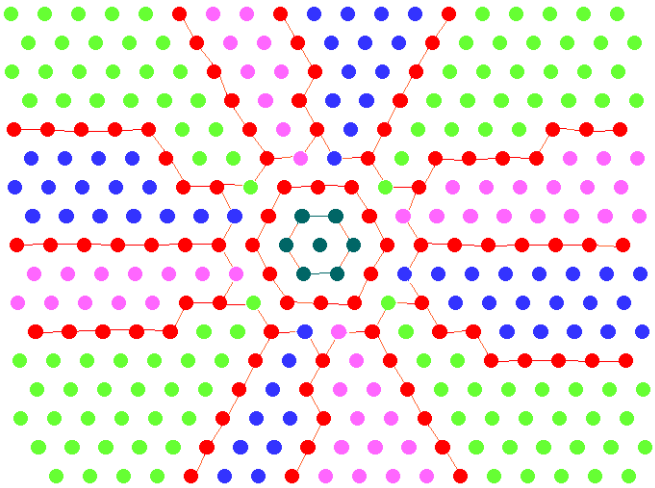
Enfin, un dernier phénomène doit alors être pris en compte et géré. Il se peut en effet qu'à la fin de l'algorithme, tous les pixels de l'image n'aient pas été traités car ils n'ont pas tous été stochés sur la file d'attente. Cela se produit notamment lorsqu'on a affaire à des configurations comme celle illustrée à la figure 11. Dans ce cas les points-collisions empêchent la propagation de l'inondation. On voit alors apparaître une "zone de partage des eaux" qui est parfaitement légitime. En effet, aucun des points non traités ne peut prétendre appartenir à un bassin versant plutôt qu'à un autre. La séparation entre bassins versants devient alors sub-pixellique. La seule solution, si on veut conserver l'isotropie de la propagation, est de considérer que ces points font partie de la LPE.



(a) Relief à deux niveaux: les points noirs sont à zéro, les points orange à 1



(b) Minima marqués par des couleurs différentes



(c) En rouge, les points-collisions, en noir, les points jamais inondés

Figure 11: Apparition de zones de partage des eaux

2ème partie: un autre algorithme isotrope

Cette deuxième partie sera consacrée à la description d'un autre algorithme de LPE isotrope qui n'est pas basée sur une file d'attente mais sur une simple transformation morphologique, la dilatation. Le but de cette présentation est double: d'une part, fournir un moyen de réaliser des LPE sans biais sur des systèmes ou des logiciels de traitement d'image classiques ne disposant pas d'une logique de FAH; d'autre part, montrer que la LPE peut se construire à l'aide de dilatations géodésiques généralisées telles qu'elles ont été introduites dans [1].

L'algorithme décrit ici a déjà été introduit dans [3,4] dans le cas restreint du squelette par zones d'influence (SKIZ). Il est également basé sur un double étiquetage pour mettre en évidence les collisions. Je le décrirai à nouveau dans le cadre du SKIZ (mais, ce n'est après tout qu'une forme particulière de LPE) avant de montrer comment il peut être utilisé, soit dans la réalisation de l'algorithme classique de LPE (utilisant le SKIZ géodésique), soit en remplaçant les dilatations par des dilatations géodésiques généralisées.

2.1) Un algorithme de SKIZ isotrope

Considérons un ensemble X constitué de n composantes connexes X_i. Nous allons construire W(X), l'ensemble des zones d'influence des composantes connexes de X en fabricant à l'aide de simples dilatations la fonction g telle que:

$$g : E \rightarrow N^+$$
$$x \rightarrow g(x) = i \text{ , } x \in IZ_W(X_i)$$

où $IZ_W(X_i)$ est la zone d'influence de la composante connexe X_i.

L'algorithme utilise un double étiquetage des composantes de X. Les étapes du traitement sont les suivantes:

- on calcule g_0:

$g_0(x) = i$ si et seulement si $x \in X_i$, $\forall i$
$g_0(x) = 0$ sinon

- on calcule alors la fonction g'_0:

$g'_0(x) = n - g_0(x)$ ssi $g_0(x) \neq 0$
$g'_0(x) = 0$ sinon

Cette fonction est en fait similaire à g_0. La différence provient de ce que les étiquettes i des ensembles X_i sont devenues (n - i).

Cette phase d'initialisation étant terminée, pour chaque g_i et g'_i, on effectue les opérations suivantes:

- calcul de la fonction h_i:

$$h_i = (g_i \oplus B) + (g'_i \oplus B)$$

B étant la boule élémentaire de E.

- on détermine alors une fonction h caractéristique de l'ensemble des points pour lesquels h_i est inférieure ou égale à n:

$$h = 1 \text{ ssi } h_i \le n$$
$$h = 0 \text{ sinon}$$

- on calcule alors les fonctions g_{i+1} et g'_{i+1}:

$$g_{i+1} = \sup[(g_i \oplus B) \times h, g_i]$$
$$g'_{i+1} = \sup[(g'_i \oplus B) \times h, g'_i]$$

L'algorithme est alors réitéré jusqu'à idempotence. On obtient finalement:

$$g = g_\infty$$

Comme le montre la figure 12, le double étiquetage initial évite un traitement individuel des composantes connexes en détectant les collisions par débordement de la fonction h_i.

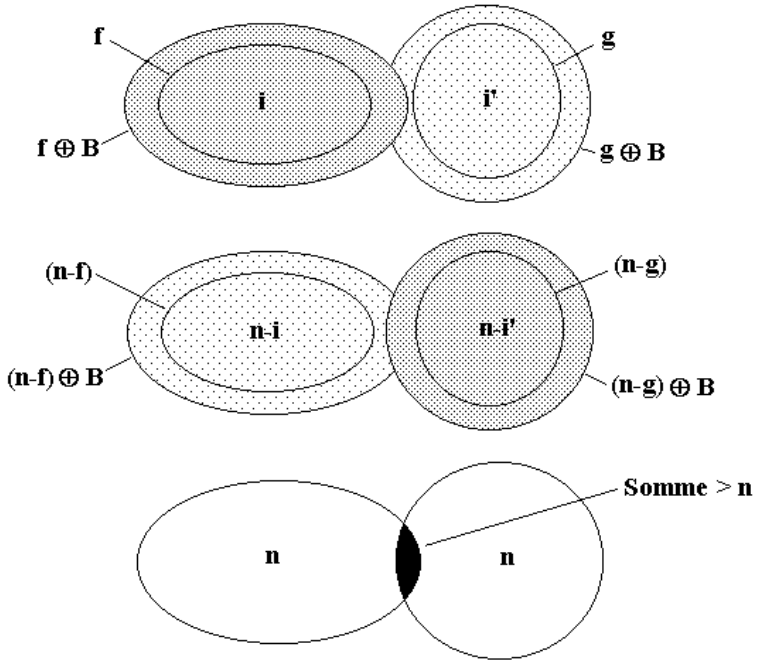


Figure 12: utilisation du double étiquetage pour détecter les collisions

On remarquera que le résultat de cet algorithme est identique à celui que l'on obtiendrait en réalisant la LPE de la fonction distance à l'aide de l'algorithme de FAH isotrope précédemment décrit. En effet, le traitement des collisions est identique. La dilatation assure une propagation isotrope de l'inondation. Le double étiquetage permet de vérifier pour chaque point atteint par la dilatation si cette dilatation provient ou non de plusieurs composantes connexes différentes. De plus, les points en collision ne sont pas dilatés, ils ne propagent donc pas de biais.

2.2) Utilisation dans la réalisation de la LPE

Comment utiliser cet algorithme pour réaliser une ligne de partage des eaux? L'approche la plus simple consiste à l'utiliser pour effectuer le SKIZ géodésique dans l'algorithme classique. Il suffit pour cela de remplacer les dilatations euclidiennes par des dilations géodésiques.

Si Z est l'espace géodésique, on a:

$$h_i = \delta_Z(g_i) + \delta_Z(g'_i)$$

Puis:

$$h = 1 \text{ ssi } h_i \leq n$$
$$h = 0 \text{ sinon}$$

et:

$$g_{i+1} = \sup[\delta_Z(g_i) \times h, g_i]$$
$$g'_{i+1} = \sup[\delta_Z(g'_i) \times h, g'_i]$$

Le SKIZ géodésique est effectué sur chaque seuil de la fonction.

On peut cependant généraliser cet algorithme en utilisant la dilatation géodésique généralisée. On trouvera en annexe A la présentation de cette notion ainsi que son utilisation pour construire la LPE. L'algorithme utilisé n'est pas plus rapide cependant que l'algorithme utilisant les seuils de la fonction.

On trouvera également à l'annexe B une réalisation de cette LPE basée sur un double étiquetage pour le logiciel MICROMORPH. Enfin, un certain nombre de résultats commentés sont présentés à l'annexe C.

3ème partie: considérations sur la parallélisation de la ligne de partage des eaux

Dans cette troisième et dernière partie, j'aborderai la question de la parallélisation de la LPE. En effet, le fait de disposer d'algorithmes isotropes amène à réfléchir sur les moyens de paralléliser la LPE puisque l'ordre de traitement des pixels n'intervenant plus, l'inondation est réellement un processus parallèle. On verra cependant qu'il existe toujours un certain nombre de contraintes à respecter absolument si on veut obtenir un résultat correct.

La parallélisation ne sera abordée qu'avec l'algorithme de FAH. Une étude comparative importante de la parallélisation de la LPE a été réalisée dans [6]. Je me baserai partiellement sur les conclusions de cette étude dans cette partie.

3.1) Parallèlisation: les contraintes importantes

Il existe plusieurs façons d'envisager la parallèlisation de la LPE par file d'attentes hiérarchiques. L'une d'elle consiste à disposer d'un certain nombre de processeurs et de faire travailler chacun d'eux en paralèle sur les pixels de l'image. Une autre consiste à découper l'image à traiter en imagettes et à associer à chaque imagette une structure de file d'attente hiérarchique complète (FAH, mémoire-label et processeur). Ces organisations amènent deux contraintes importantes:

- une contrainte au niveau du séquencement des processeurs et des files d'attente.
- une contrainte au niveau de l'échange d'information entre les structures concernant les pixels ou jetons à la frontière des imagettes.

L'approche à l'aide d'imagettes présente l'avantage que l'architecture de se chaque structure de FAH peut être calibrée pour une taille d'imagette donnée et qu'il suffit de

rajouter autant de structures qu'il est nécessaire lorsqu'on veut traiter des images d'une taille donnée. Dans la suite, nous nous intéresserons principalement à cette approche.

3.1.1) Le séquencement des processeurs

Il est fort probable que toutes les FAH associées aux différentes imagettes ne contiendront pas au même moment le même nombre de jetons. Il est donc indispensable de veiller à ce que les différentes structures travaillent en cadence. Cela signifie en particulier que chaque file d'attente de priorité courante ne pourra pas être traitée tant que toutes les files d'attente de priorité supérieure dans toutes les FAH n'auront pas été vidées. Cette règle est valable pour toutes les files d'attentes et en particulier pour toutes les files α telles qu'elles ont été définies précédemment. On peut aussi illustrer le fonctionnement des files d'attente en considérant que la file d'attente courante est munie d'un blocage ou d'une cheville (figure 13) empêchant les jetons empilés au cours du traitement d'être traités tant que tous les jetons de génération antérieure ne sont pas sortis de la pile.On rappelle également qu'il ne peut y avoir à tout instant du traitement qu'un seul taquet dans la file courante (ou, ce qui revient au même, qu'une seule file α).

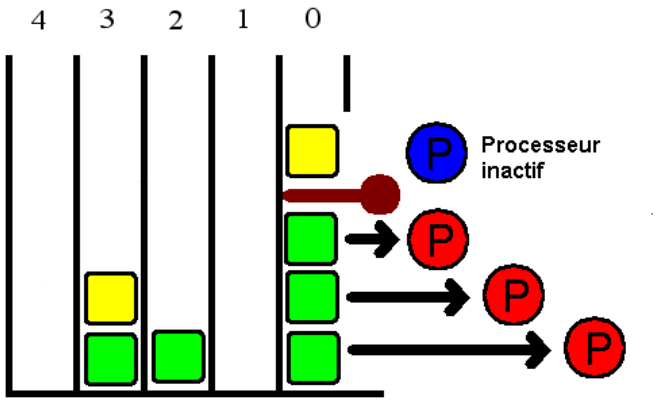


Figure 13: principe du blocage du traitement des jetons de générations différentes
Le jeton jaune de la file d'attente de priorité 0 ne sera dépilé (blocage ôté)
que lorsque tous les jetons de génération antérieure dans toutes les FAH
auront été traités

Chaque file d'attente associée à chaque imagette doit donc indiquer (par un drapeau par exemple) son statut. Elle doit également vérifier avant de commencer le traitement de la génération et/ou priorité suivante de jetons que tous les jetons de génération et/ou priorité courante ont été traités. Elle le fait en contrôlant l'état des drapeaux de toutes les FAH.

Deux questions se posent: premièrement, est-il indispensable d'installer ce système de blocage sur toutes les files ou suffit-il de le faire sur la file courante? Deuxièmement, est-on sûr de ne pas empiler sur la file courante des jetons de générations n+1, n+2, etc. pendant le traitement des jetons de génération n? La réponse à ces deux questions est évidente lorsqu'on sait que la propagation ne se fait que dans la file courante d'une part et que pour qu'un jeton de génération n+2 apparaisse, il faut qu'un jeton de génération n+1 soit traité, ce qui n'est pas le cas lorsqu'on traite les jetons de génération n. Il suffit donc d'installer le mécanisme de contrôle sur la file d'attente courante.

On serait tenté de croire que, lors de la construction de la LPE par un algorithme parallèle utilisant un découpage de l'image, une vigilance s'impose seulement au bord des

imagettes lorsque la propagation risque de passer dans une imagette adjacente et que, en dehors de ces situations, le fonctionnement de la FAH pour tous les pixels intérieurs de l'imagette peut se faire sans se préoccuper de ce qui se passe dans les autres imagettes. Cette assertion (qui n'a pas manqué d'être reprise maintes fois) est *totalement fausse* et conduit à des résultats calamiteux! On peut l'illustrer très simplement à l'aide d'un exemple simple (figure 14). On voit que, parce que la frontière entre les deux imagettes ne passe pas au milieu du plateau, la position de la LPE sera totalement erronée.

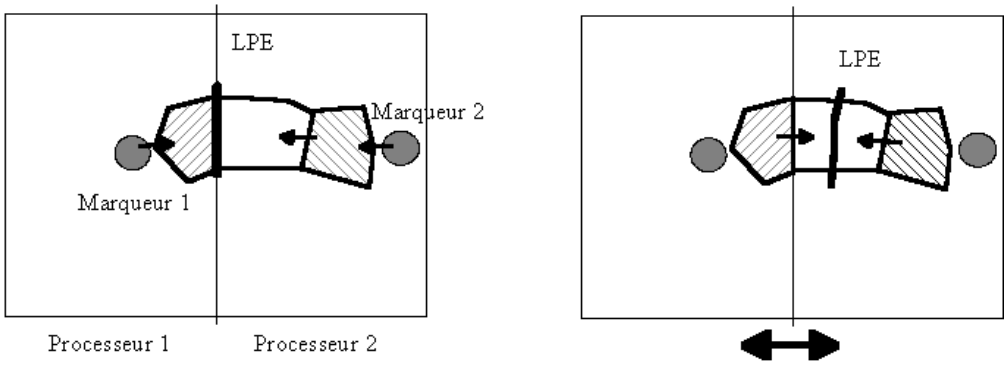


Figure 14: biais dans la LPE parallèle si on ne se préoccupe que des pixels à la bordure des imagettes adjacentes

3.1.2) L'échange d'information entre processeurs

Dans cette structure parallèle, chaque processeur dispose de sa propre imagette, qu'il va lire, de sa propre image-label qu'il va lire et écrire ainsi que de sa propre file d'attente. Tant que les pixels traités ne sont pas situés sur le bord des imagettes et tant qu'on respecte le fonctionnement en cadence de toutes les files d'attente, il n'y a pas de problème puisque les accès ne peuvent se faire que dans les mémoires associées à chacun des processeurs. Par contre, lorsque le pixel est au bord de l'imagette, certains pixels voisins appartiendront à des imagettes adjacentes et devront normalement être empilés dans les FAH associées aux processeurs traitant ces imagettes. Cela n'étant pas possible (il ne peut y avoir double accès à une adresse-mémoire), un autre mécanisme d'échange doit être utilisé. Pour ce faire, une mémoire tampon est associée à chaque imagette adjacente à l'imagette courante. Il y a donc pour chaque structure imagette/FAH/image-label A un tampon noté AB pour chaque structure adjacente B. Ce tampon peut contenir l'information associée à un jeton.

Considérons deux structures processeur/FAH gérant deux imagettes adjacentes A et B. Soit alors un jeton appartenant à l'imagette A et adjacent à B. Ce jeton en sortie de pile va être pris en charge par le processeur A qui va propager l'inondation vers tous ses points voisins inclus dans l'imagette A. Mais ce jeton sortant est également stocké dans le tampon AB. Dans le même temps, il positionne un drapeau indiquant au processeur adjacent que le tampon est plein. Ce drapeau agit comme une interruption pour le processeur adjacent B qui va arrêter sa tâche courante (dès que le traitement de son jeton courant sera terminé) et prendre en charge le jeton stocké dans le tampon comme s'il sortait de sa propre file d'attente. Les voisins de ce jeton appartenant à l'imagette B seront pris en compte normalement par le processeur B.

Pour compléter le fonctionnement, d'autres mécanismes de contrôle doivent être mis en place. Ainsi, plusieurs tampons peuvent se remplir en même temps. Il faut donc prévoir un traitement séquentiel de ces tampons (l'ordre de traitement est indifférent). De la même

façon, si un tampon n'est pas vide lorsqu'un processeur tente de le remplir, ce dernier doit attendre qu'il se vide.

L'ordinogramme suivant résume le fonctionnement du processeur k (le contrôle de la concordance des générations/files actives entre processeur n'est pas inclus dans cet ordinogramme).

{Pour chaque tampon km (m, processeur adjacent), faire
{Si tampon km plein, faire:
{traiter le jeton stocké dans le tampon}
{vider le tampon}
}
}
{Dépiler le jeton courant de la FAH k}
{Si le jeton est adjacent au processeur m, faire
{Si tampon mk plein, attendre la fin de son traitement par le processeur m}
{Mettre le jeton dans le tampon mk}
}
{Sinon, le jeton est pris en charge normalement par le processeur k}

3.2) L'efficacité de la parallèlisation

Comme on peut s'en douter, la parallèlisation de la FAH n'est pas un processus à haut rendement. En effet, pour assurer la réalisation d'une LPE sans biais, de nombreux contrôles doivent être élaborés, ce qui pénalise le rendement de la parallèlisation. De plus, les structures d'échange des jetons doivent être prévues pour toutes les imagettes adjacentes, y compris celles situées en coin qui ne peuvent au plus s'échanger qu'un jeton. Cependant, le négliger pourrait entraîner des biais considérables. Il est donc fort rare que l'utilisation de N processeurs divise par N le temps de traitement. De plus, il est probable qu'il existe un nombre optimal de processeurs: trop petit, les gains de vitesse sont trop faibles par rapport à la complexité accrue du traitement, trop important, la communication entre processeurs risque de ralentir considérablement l'algorithme. Enfin, la géométrie du découpage des imagettes, qui conditionne la topologie des tampons, doit aussi être prise en considération.

Actuellement, on peut même se demander si la meilleure solution ne consiste pas à miser sur l'accroissement continuel des capacités de calcul des processeurs traditionnels. La question reste en suspens.

Conclusion générale

En guise de conclusion, j'aimerais rappeler les motivations principales de ce papier. Depuis une dizaine d'années, de nombreuses tentatives ont été faites pour augmenter la vitesse de la LPE. Ces tentatives ont souvent été couronnées de succès, même si ce succès est souvent dû à l'augmentation continuelle des performances des processeurs plutôt qu'à l'utilisation d'algorithmes nouveaux et plus efficaces que la file d'attente hiérarchique qui reste encore une des architectures les plus performantes en la matière. Cependant, cet augmentation de la vitesse n'a pas été accompagnée d'une augmentation de la qualité. Pire, elle s'est même souvent faite au détriment de celle-ci. Je passerai volontairement sous silence

les nombreuses propositions d'algorithmes (mixtes, parallèles, etc.) parues dans la littérature ces dernières années produisant des LPE plus que douteuses... On peut s'interroger sur les raisons qui ont fait que cette amélioration de la qualité de la LPE a, semble-t-il, rarement fait partie des préoccupations des utilisateurs. Ces raisons sont diverses. La première (et sans doute la plus pernicieuse...) est l'ignorance de l'existence d'un problème. Il suffit de proposer un algorithme qui, bien que ne respectant pas les règles incontournables de construction de la LPE, fournisse un résultat ayant l'apparence d'une ligne de partage des eaux pour s'en contenter et éviter d'entrer dans une analyse plus approfondie. Il est vrai qu'il est souvent difficile avec une image réelle de savoir quelle devrait être la "vraie" LPE. La seconde raison provient, tout en sachant qu'un biais existe, de la croyance en la ténuité de ce biais. Les divers exemples illustrant ce papier montrent tout le crédit que l'on peut donner à ce genre d'assertion. La troisième raison est dûe certainement à l'utilisation même de la LPE. Pendant longtemps, cette transformation était considérée comme l'étape finale d'un processus de segmentation. Il n'était pas critique alors d'avoir une LPE sans biais. Si l'utilisateur estimait que la segmentation était satisfaisante, d'éventuelles erreurs lui importaient peu. Mais, actuellement, la LPE apparaît de plus en plus comme un opérateur initial utilisé dans des procédures de plus en plus complexes (segmentation hiérarchique, multi-canaux, multi-critères, etc.). On est souvent amené à assigner des attributs aux différents bassins versants et à comparer ces attributs entre bassins versants adjacents. On peut aussi avoir à comparer des LPE ou des portions de LPE. Il est alors certain que, dans ces cas, les biais pénalisent fortement la robustesse des algorithmes. Il devient donc urgent de faire de la LPE un outil fiable et reproductible. En résumé, il est urgent de faire passer la LPE du statut d'outil complexe à celui d'outil morphologique de base. Ce changement de statut risque d'accroître encore les exigences des utilisateurs quant aux performances. C'est pourquoi j'ai tenu à montrer qu'on peut parallèliser la LPE tout en lui conservant de bonnes propriétés d'exactitude. L'algorithme présenté dans la troisième partie, bien que succinct, essaie de respecter les contraintes propres à la parallèlisation, à l'accès à l'information. J'espère qu'il incitera les concepteurs d'architecture à se pencher dessus pour en fournir à terme une vraie réalisation matérielle.

Références

[1] Beucher S. Segmentation d'Images et Morphologie Mathématique. Thèse de Doctorat, Ecole des Mines de Paris, Juin 1990.

[2] Beucher S. Watershed, hierarchical segmentation and waterfall algorithm. Proc. Mathematical Morphology and its Applications to Image Processing, Fontainebleau, Sept. 1994, J. Serra and P. Soille (Eds), Kluwer Ac. Publ., Nld, 1994, pp. 69-76.

[3] Beucher S. Interpolations d'ensembles, de partitions et de fonctions. Note interne CMM n° N-18/94/MM, Mai 1994 (confidentiel).

[4] Beucher S. Sets, partitions and functions interpolations. Mathematical Morphology and its Applications to Image Processing. Proc. ISMM'98, Amsterdam, June 1998, H.J.A.M. Heijmans and J.B.T.M. Roerdink (Eds) , Kluwer, Dordrecht, 1998, pp. 307-314.

[5] Beucher S. Watersheds & waterfalls. Paris School of Mines, “Vision and Morphology” course, Feb. 2000.

[6] Beucher S, Colin T, Kajfasz P, Lemonnier F, Sasportas R. Réalisation de la ligne de partage des eaux par file d’attente hiérarchique parallèle - Etude d’architecture, modélisation sous Ptolemy. Rapport final, contrat DGA-DRET n° 95-520, Octobre 1997 (confidentiel).

[7] Beucher S, Meyer F. The morphological approach of segmentation: the watershed transformation. In Dougherty E. (Editor), Mathematical Morphology in Image Processing, Marcel Dekker, New York, 1992.

[8] Serra J. Image Analysis and Mathematical Morphology, Academic Press, London, 1982.

Annexe A

La dilatation géodésique généralisée

La généralisation de la distance géodésique consiste à calculer les distances entre deux points en termes de *temps de parcours minimal* et non plus simplement de longueur de chemin minimal.

Reprenant la définition de la distance géodésique, on sait que cette distance est égale à la longueur du chemin minimal entre deux points. Soit C_{xy} , un chemin quelconque entre deux points x et y d'un ensemble X connexe. Sa longueur est l'intégrale curviligne:

$$L(C_{xy}) = \int_{C_{xy}} ds$$

ds est l'élément linéaire élémentaire découpé sur le chemin C_{xy} . Cette longueur peut également être exprimée en temps de parcours. L'élément ds est parcouru en un temps dt proportionnel à sa longueur:

ds = v dt, v est la vitesse de parcours

On a alors :

$$L(C_{xy}) = v \int_{C_{xy}} dt = v\, T(C_{xy})$$

Comme v est constant, le temps de parcours total $T(C_{xy})$ du chemin peut être pris en compte pour mesurer sa longueur. La distance entre x et y est donc égale en temps de parcours minimal entre x et y.

Cependant, il se peut que la vitesse de déplacement ne soit pas constante mais varie en chaque point de l'espace en fonction de sa position mais également selon l'orientation de l'élément ds.

En désignant par $\omega(s)$ cette vitesse à l'abscisse curviligne s le long du chemin C_{xy}, sa longueur exprimée en temps de parcours devient:

$$T(C_{xy}) = \int_{C_{xy}} \frac{ds}{\omega(s)}$$

Désignons par η, l'inverse de la fonctionnelle ω:

$$\eta = 1/\omega$$

On a alors:

$$T(C_{xy}) = \int_{C_{xy}} \eta(s)\, ds$$

Cette fonctionnelle η est appelée *réfringence* par analogie avec la propagation lumineuse. En effet, la vitesse de déplacement est d'autant plus grande que η est petit. Cette fonctionnelle dans le cas d'un milieu homogène peut être représentée par un tenseur car sa valeur varie avec la direction.

Dans le cas digital, la réfringence η peut être représentée comme une valuation des arêtes du graphe (Z, D) de digitalisation: Z est l'ensemble des sommets du graphe et D, l'ensemble des arêtes. On peut alors associer à chaque arête $x_i x_j \in D$ une valeur numérique positive ou nulle η_{ij} représentant la réfringence ou le temps de parcours entre x_i et x_j. Dans ce cas, cette valuation est une distance ou au moins un *écar*t si on s'autorise des valeurs nulles.

On peut aussi associer au point x un temps de parcours associé à chaque direction i définie sur le graphe, supposé régulier (ou à chaque voisin du point x). Cette représentation est encore plus générale puisqu'elle permet de définir des temps de parcours dissymétriques.

La réfringence η est alors un vecteur à n composantes, n étant le nombre de voisins d'un point dans le graphe:

$$\eta = (\eta_1, ..., \eta_i, ..., \eta_n)$$

où $\eta_i(x)$ représente au point x le temps de parcours de l'arête d'origine x et orientée dans la direction i.

On peut alors définir la dilatation géodésique d'ensemble selon cette réfringence. Soit Y l'ensemble à dilater, et t la taille en unités de temps de cette dilatation ($t \in N$). Le principe de l'algorithme de dilatation est le suivant:

Considérons une des fonctions η_i et sa restriction à l'ensemble Y. Tous les points x de Y tels que $\eta_i(x) = 0$ sont dilatés dans la direction i. En effet, les points du dilaté sont des points, dans la direction i, à une distance (ou écart) nulle de Y. La même opération doit être effectuée dans toutes les directions i et réitérée jusqu'à idempotence afin d'adjoindre à Y tous les points dont l'écart à Y est nul. On obtient alors un nouvel ensemble Y'. Evidemment, si η représente une distance, Y' est égal à Y. On calcule alors pour chacune des fonctions η_i pour tous les points x intérieurs à Y' la valeur :

$$\sup(\eta_i - 1, 0)$$

On produit alors une nouvelle fonction η'_i, égale à η_i pour tous les points extérieurs à Y', et dont la valeur a été decrémentée de 1 (jusqu'à 0) pour tous les points intérieurs à Y'. Comme précédemment, tous les points x de Y' tels que $\eta'_i(x) = 0$ sont les points pour lesquels le point adjacent dans la direction i est à la distance unité (en temps de parcours). La dilatation de ces points adjoint donc à Y' les points à la distance 1. Pour effectuer une dilatation de taille t, il suffit d'itérer cette procédure élémentaire.

Cet algorithme utilise des dilatations linéaires. Afin de l'accélérer, on peut légèrement le modifier. On commence par modifier la définition des fonctions η_i, composantes de la réfringence. La valeur $\eta_i(x)$ au point x représentera désormais le temps de parcours nécessaire pour atteindre le point x en venant de la direction conjuguée à i (les nouvelles fonctions sont obtenues en fait par un simple décalage des précédentes). On peut alors définir un ensemble Z_0:

$$Z_0 = \bigcup_i \{x : \eta_i(x) = 0\}$$

Z_0 représente l'ensemble des points pour lesquels il existe un point adjacent à une distance nulle. Soit Z, l'ensemble formé par l'union de Y et de Z_0:

$$Z = Y \cup Z_0$$

On construit alors l'ensemble Y', ensemble des points à une distance nulle de Y à l'aide d'une reconstruction géodésique de Z par l'ensemble marqueur Y:

$$Y' = R_Z(Y)$$

L'intérêt de cette variante est qu'il n'est plus nécessaire d'effectuer des dilatations linéaires. On modifie alors les fonctions η_i:

$$\eta'_i(x) = \sup(\eta_i(x) - 1, 0) \text{ si } x \in Y' \oplus B$$
$$\eta'_i(x) = \eta_i(x) \text{ sinon.}$$

On détermine à nouveau Z_0 et Z:

$$Z_0 = \bigcup_i \{x : \eta'_i(x) = 0\} \; ; \; Z = Y' \cup Z_0$$

Et finalement, le dilaté $\delta_\eta(Y)$ de taille 1 (en temps de parcours) de Y s'obtient par une dilatation géodésique classique de Y' dans Z:

$$\delta_\eta(Y) = \delta_Z(Y')$$

La figure A1 illustre cette dilatation géodésique généralisée basée sur une réfringence η (symétrique dans ce cas).

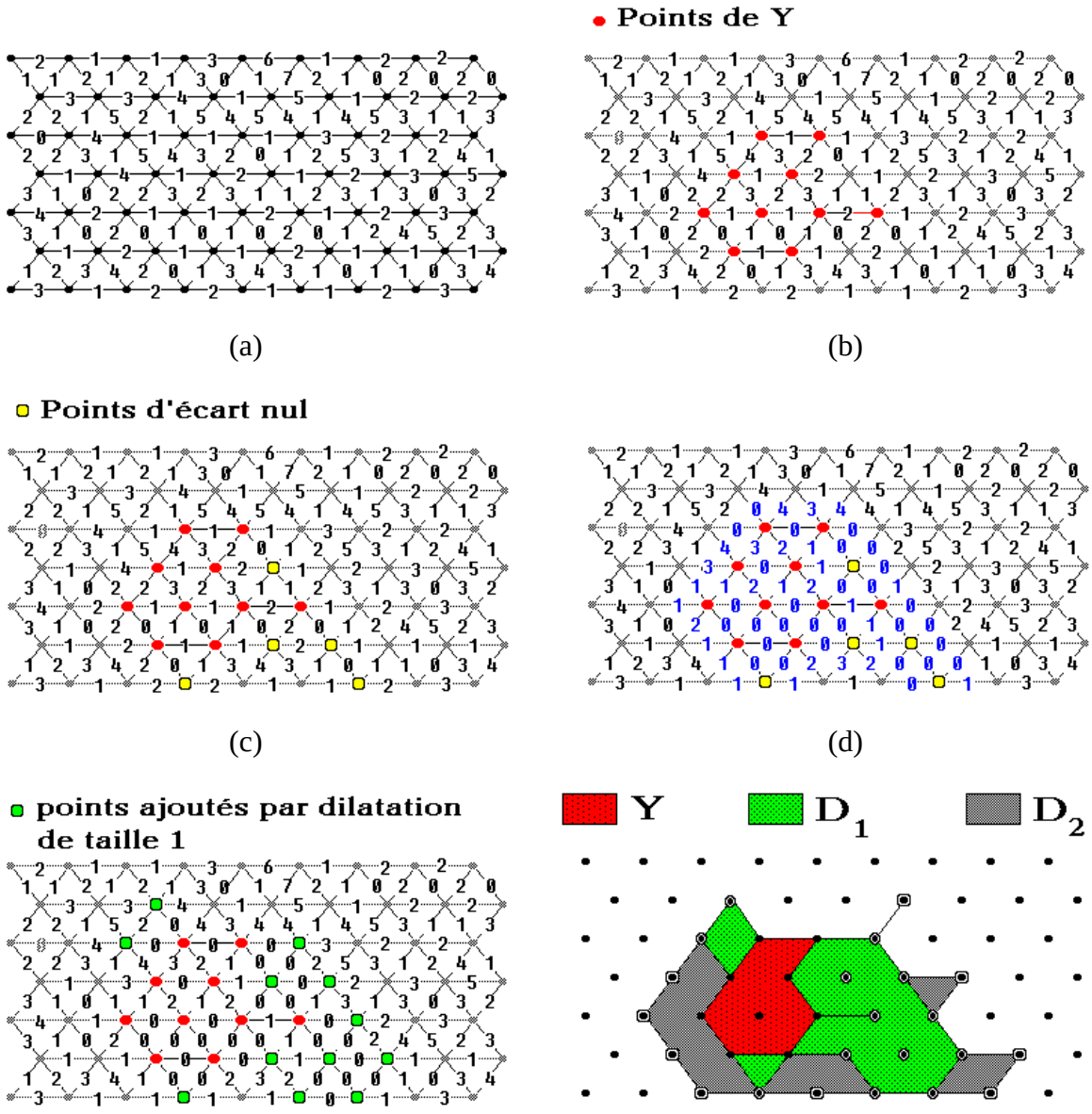


Figure A1: dilatation géodésique généralisée
(a) réfringence, (b) ensemble Y à dilater, (c) points d'écart nul
(d) réfringence modifiée, (e) dilaté unitaire, (f) dilatés de taille 1 et 2

L'érosion peut se définir par dualité. Notons que ces transformations s'étendent sans difficulté à des fonctions.

Ligne de partage des eaux et distance généralisée

Comme on l'a déjà montré [1], la LPE n'est qu'un SKIZ utilisant une fonction distance généralisée basée sur une réfringence bâtie à partir de la fonction f. Si M est l'ensemble des marqueurs et f, une fonction qu'on supposera strictement positive, on

commence par mettre à zéro les points de f correspondant aux marqueurs. Chaque fonction η_i (i est la direction) est alors définie de la façon suivante en chaque point x:

$$\eta_i(x) = \sup(f(x) - f(y), 0)$$

y étant le point adjacent à x dans la direction conjuguée à i.

En effet, le temps de propagation de l'inondation est proportionnel à la différence d'altitude entre les points adjacents, pourvu que cette différence soit positive. Si la différence est négative, le temps de propagation est nul. Cela permet, en quelque sorte, d'intégrer dans la définition de la réfringence la modification d'homotopie.

On peut alors réaliser le SKIZ géodésique selon cette distance en utilisant le double étiquetage précédemment décrit en veillant à contrôler la propagation sur les plateaux en décomposant chaque étape de cette propagation à l'aide de dilatations géodésiques élémentaires classiques.

Cet algorithme n'est pas plus rapide que l'algorithme classique basé sur les seuils de la fonction f. Sa présentation a pour seul but d'éclairer différemment la définition et la réalisation de la LPE.

Annexe B

Cette annexe contient le programme commenté de la LPE sans biais présenté dans la deuxième partie de ce papier. Il est écrit à l'aide du logiciel MICROMORPH. Les lecteurs intéressés par une version électronique de ce programme peuvent le demander à l'auteur.

Procédure dblabel

Double étiquetage de l'image binaire s. Les images numériques d1 et d2 contiennent respectivement les composantes connexes de s étiquetées dans l'ordre du balayage avec des labels n et 255-n. Aucun contrôle du nombre de composantes connexes n'est réalisé au cours du traitement (il faut vérifier que ce nombre est inférieur à 255 avant de lancer la procédure).

```
deproc dblabel dblabel s d1 d2
syntax "double étiquetage de s"
int w v u i;
  w := imalloc 1
  v := imalloc 1
  u := imalloc 8
  i := 1
  imcopy s w
  imset 0 d1
  while (imvolume w <> 0) do
    fgrain w v
    immask v 0 i u
    imsup d1 u d1
    i := (i + 1)
  end
  immask s 0 255 u
  iminv d1 d2
  iminf d2 u d2
  imfree w
  imfree v
  imfree u
end
```

Procédure izcontour

Cette procédure extrait les contours des bassins versants ou des zones d'influence étiquetés de l'image imin et stocke le résultat dans imout.

Attention! Cette procédure n'a pour but que de permettre de comparer les résultats obtenus avec l'algorithme isotrope avec ceux produits par l'algorithme classique utilisant les épaississements. Cependant, les contours obtenus peuvent être d'épaisseur 1 ou 2, ce qui peut entraîner des biais: bassins versants déconnectés voire même éliminés. Il est donc fortement déconseillé d'utiliser l'image fournie par cette procédure dans des traitements ultérieurs.

```
deproc izcontour izcontour imin imout
syntax "contours des BVs"
int w w1 w2;
  w := imalloc 1
  w1 := imalloc 8
```

```
  w2 := imalloc 8
  imthresh imin 1 255 w
  gdsdil imin w w1 1
  gdsero imin w w2 1
  imsub w1 w2 w1
  imthresh w1 1 255 imout
  iminv w w
  imsup imout w imout
  imfree w
  imfree w1
  imfree w2
end
```

Procédure tgdskiz

Squelette par zone d'influence géodésique. Les images im1 et im2 doivent contenir les composantes connexes étiquetées par la procédure dblabel ci-dessus. L'image mask contient l'espace géodésique. Le résultat du SKIZ se trouve sous la forme d'images étiquetés des zones d'influence stockées dans im1 et im2 (bien que les labels soient différents, ces deux images sont formellement identiques).

```
deproc tgdskiz tgdskiz im1 im2 mask
syntax "SKIZ géodésique avec deux images-label"
int wd w1 w2 w v i;
  wd := imalloc 16
  w1 := imalloc 8
  w2 := imalloc 8
  w := imalloc 8
  v := imalloc 1
  i := 0
  while (imvolume im1 > i) do
    i := (imvolume im1)
    dil im1 w1 1
    dil im2 w2 1
    imadd w1 w2 wd
    imcopyhibyte wd w
    imthresh w 0 0 v
    iminf mask v v
    gandb w1 v w1
    gandb w2 v w2
    imsup im1 w1 im1
    imsup im2 w2 im2
  end
  imfree wd
  imfree w1
  imfree w2
  imfree w
  imfree v
end
```

Procédure trueskiz

SKIZ non biaisé de l'image binaire imin. Le résultat dans imout est l'image des labels des différentes zones d'influence.

```
deproc trueskiz trueskiz imin imout
syntax "SKIZ non biaisé"
int w v;
  w := imalloc 8
  v := imalloc 1
  dblabel imin imout w
  imset 1 v
  tgdskiz imout w v
  imfree w
  imfree v
end
```

Procédure truewshed

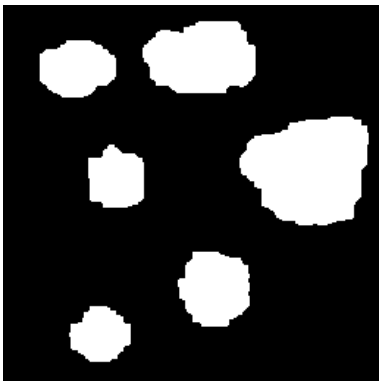
Cette procédure fournit la ligne de partage des eaux isotrope de l'image numérique imin. Cette LPE est contrôlée par les marqueurs binaires situés dans l'image m. L'image imout contient les bassins versants étiquetés.

```
deproc truewshed truewshed imin m imout
syntax "LPE isotrope contrôlée par marqueurs"
int w w1 w2 j;
  w := imalloc 1
  w2 := imalloc 8
  dblabel m imout w2
  j := 0
  for 0 to 255 do
    imthresh imin 0 j w
    imsup w m w
    tgdskiz imout w2 w
    j := (j + 1)
  end
  imfree w2
  imfree w
end
```

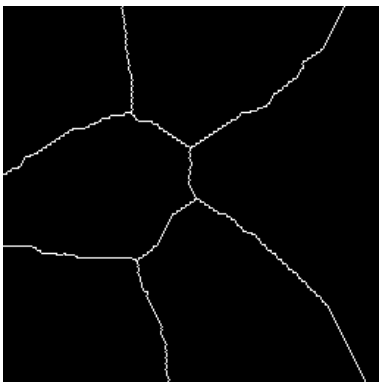
Annexe C

Images-résultats commentées

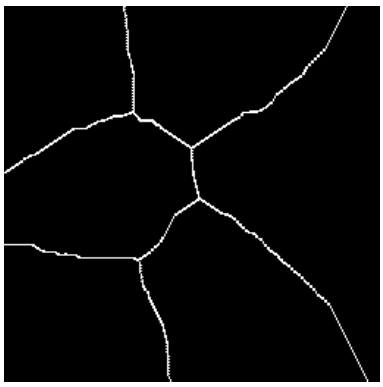
Voici, pour illustrer les propos de cet article, deux exemples de résultats. Le premier est une comparaison des différents squelettes par zones d'influence obtenus soit avec l'algorithme classique, soit avec l'algorithme isotrope, en trame hexagonale ou carrée. Le second exemple est la ligne de partage des eaux de l'image d'électrophorèses (incontournable, n'est-ce pas!).



(a)

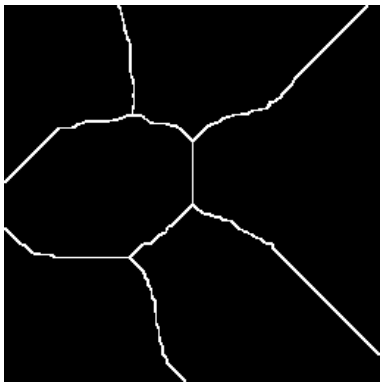


(b)



(c)

(d)

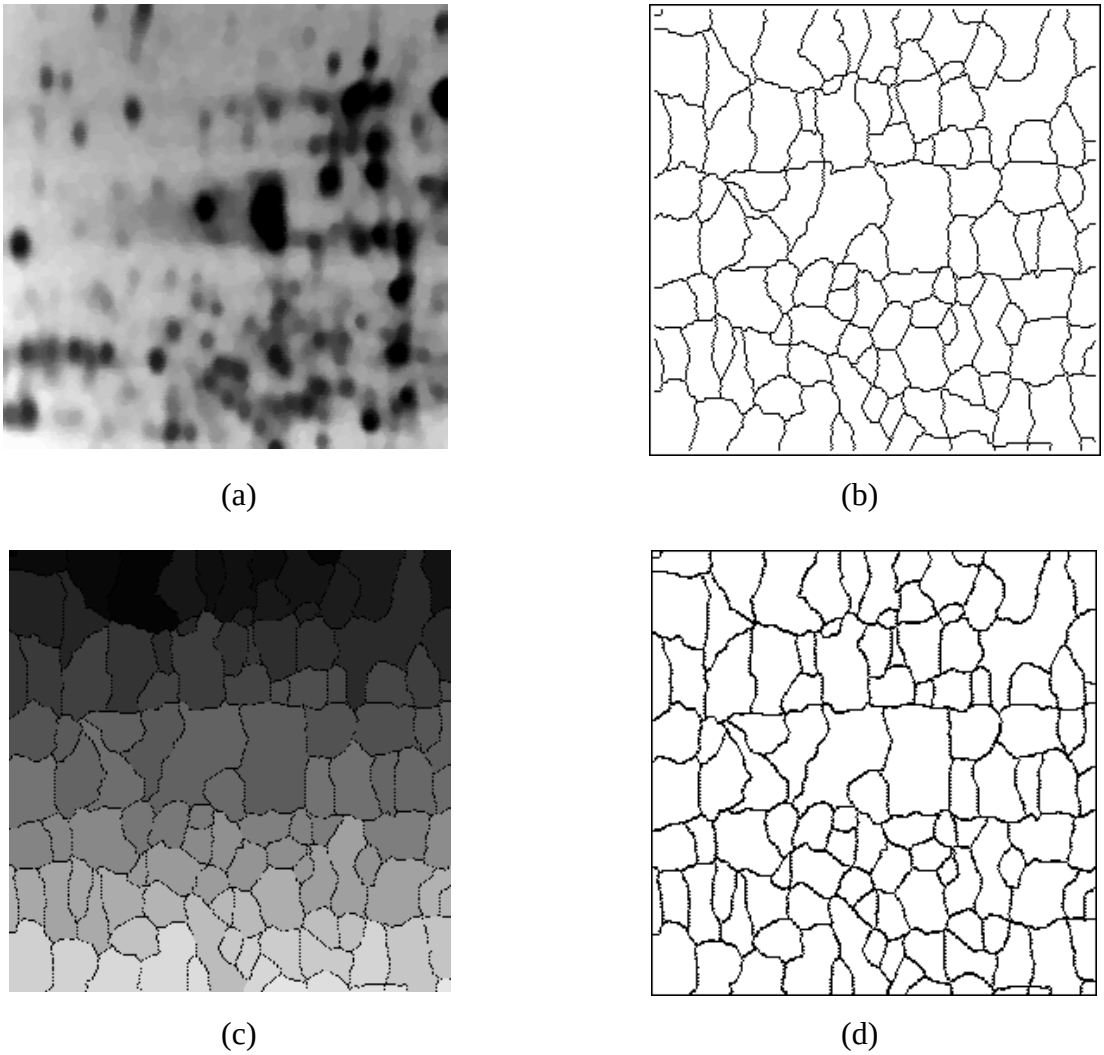


(e)

Figure C1: ensemble original (a), SKIZ hexagonal classique (b), SKIZ hexagonal vrai (c) SKIZ en trame carrée classique (d), SKIZ carré vrai (e)

Cet exemple montre que le SKIZ en trame carré classique est particulièrement biaisé. Ce biais est moins sensible en trame hexagonale dans cet exemple. On remarquera que, dans les deux cas, le SKIZ isotrope est néanmoins dépendant de la distance utilisée implicitement pour le réaliser. On voit notamment que, loin des composantes connexes, le pouvoir de séparation des distances s'atténue et que le SKIZ a tendance à suivre les directions principales de la trame.

Le second exemple (figure C2) montre les lignes de partage des eaux de l'image d'électrophorèse avec l'algorithme classique et avec l'algorithme isotrope en trame hexagonale.



(a) (b)

(c) (d)

Figure C2: image originale (a), LPE obtenue avec l'algorithme classique (b) Bassins versants de la LPE isotrope (c), contours des bassins de la LPE isotrope (d)

Dans ce deuxième exemple, le phénomène le plus intéressant est le biais considérable qui apparaît pour certains bassins versants, notamment dans la région désignée dans la figure C3 (zone entourée en rouge). Une erreur ponctuelle dans la propagation de l'inondation dans la LPE classique fait que celle-ci s'est propagée dans le mauvais bassin versant engendrant ainsi une "fuite" spectaculaire.

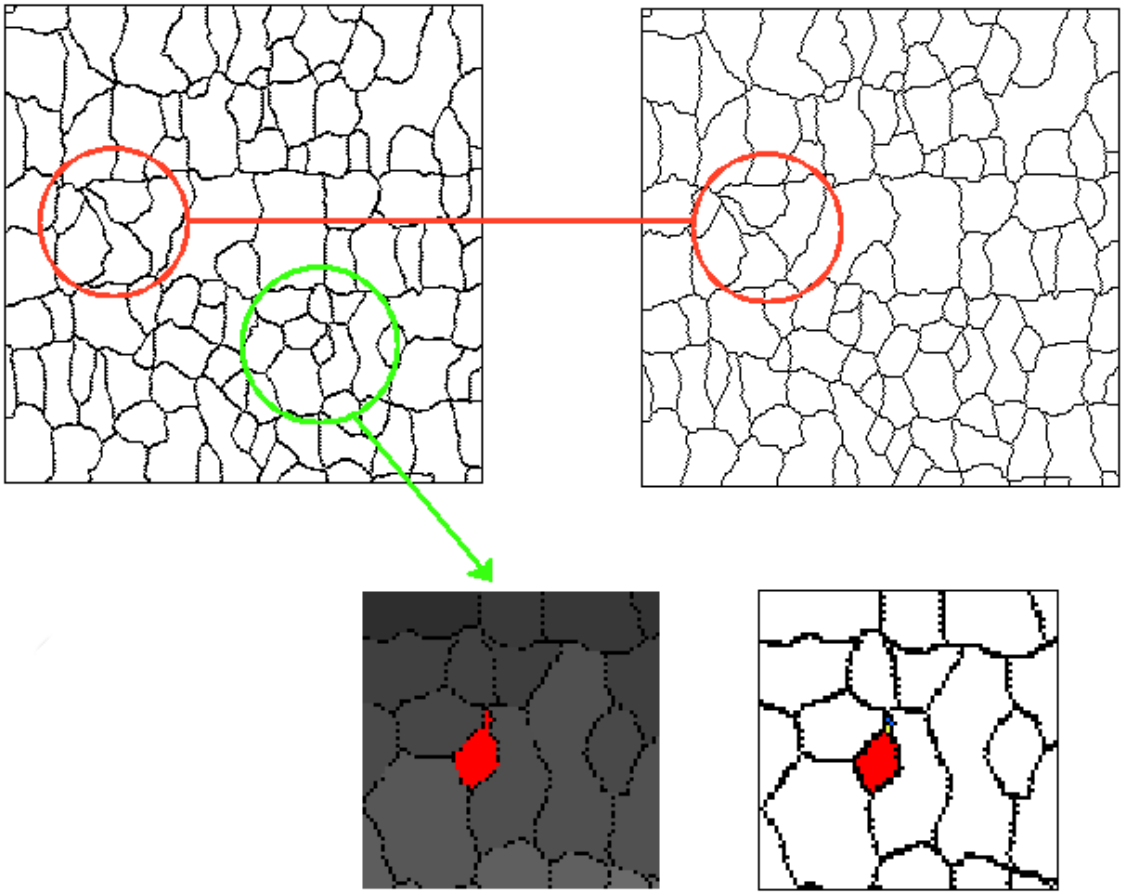


Figure C3: erreur dans la propagation avec la LPE classique
(dans la région entourée en rouge).
Dans la zone verte, déconnexion d'un bassin versant lors du tracé de ses contours

On remarquera aussi que la mise en évidence des contours des bassins versants de la LPE sans biais peut, dans certains cas, déconnecter ces bassins versants, voire les supprimer. C'est le cas, en particulier pour le bassin versant situé dans la zone entourée de vert dans la figure C3.

Ces exemples montrent la réalité et l'ampleur des biais générés par la LPE classique. Rappelons enfin que ces biais sont encore plus prononcés pour les images 3D.